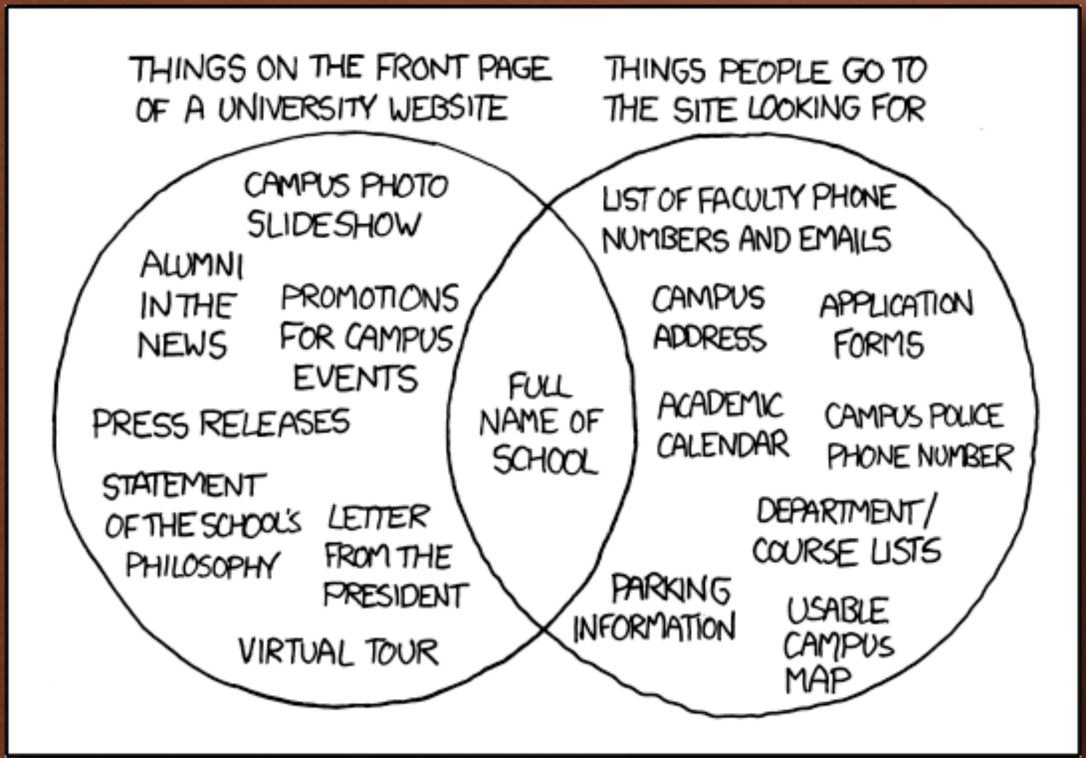
THE BUILDING BLOCKS: MAKING FUNCTIONS FROM GATES



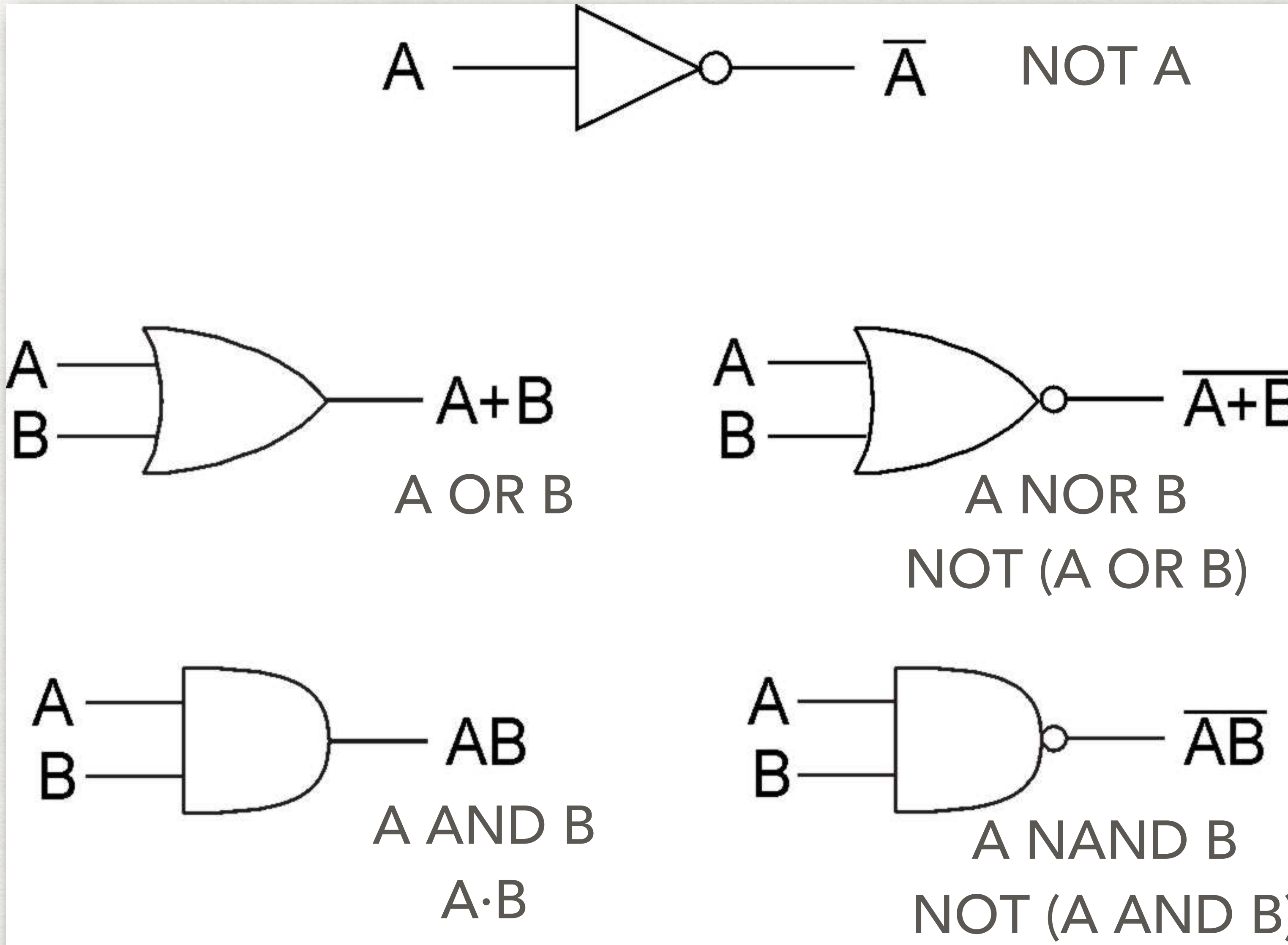
https://xkcd.com/773/

LEARNING OBJECTIVES

- Summary of gates
- Showing that electronics is not necessary for logic gates
- Equivalence of circuits and boolean expressions
- Constructing circuits from boolean expressions (and hence creating functions)
- Constructing circuits from truth tables using the sum of products algorithm

STILL CHAPTER 4

BASIC GATES



$\overline{A \cdot B}$

NOT JUST WIRES

- We can implement gates in many different ways, here are just a few - the only thing special about electronics is that we can make gates smaller and faster
- Redstone in Minecraft - https://www.youtube.com/watch?v=OOWaYNf35X4
  - and also without redstone - https://www.youtube.com/watch?v=753sKN64YhQ
- in Lego - https://www.youtube.com/watch?v=SYi9sJkS19Q
- as a sculpture (in our building) - https://www.youtube.com/watch?v=GQ_JZgj9o9w (starting at 1 minute)

OR FROM AND AND NOT

A	B	not A	not B	not A and not B	not (not A and not B)
0	0	1	1	1	0
0	1	1	0	0	1
1	0	0	1	0	1
1	1	0	0	0	1

so not (not A and not B) is the same as A or B $\overline{\overline{A} \bullet \overline{B}} = A + B$

this means (not A and not B) is the same as not (A or B) $\overline{A} \bullet \overline{B} = \overline{A + B}$

which is A nor B

This is one form of DeMorgan's law

AND FROM OR AND NOT

A	B	not A	not B	not A or not B	not (not A or not B)
0	0	1	1	1	0
0	1	1	0	1	0
1	0	0	1	1	0
1	1	0	0	0	1

so not (not A or not B) is the same as A and B $\overline{\overline{A} + \overline{B}} = A \bullet B$

this means (not A or not B) is the same as not (A and B) $\overline{A} + \overline{B} = \overline{A \bullet B}$

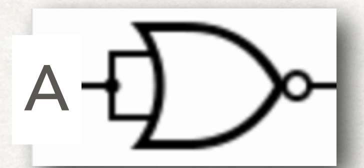
which is A nand B

This is another form of DeMorgan's law

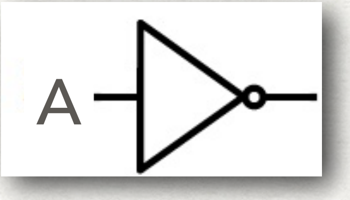
NOT FROM NOR

- If we use the same value as the two inputs to NOR we get NOT

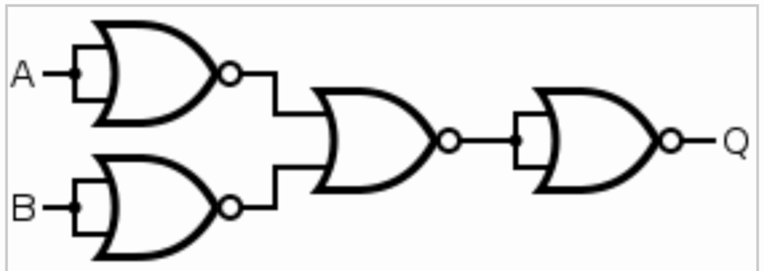
A	A	A nor A
0	0	1
1	1	0



same as



- We can then make a NAND gate from a NOR gate



NAND gate constructed using only NOR gates

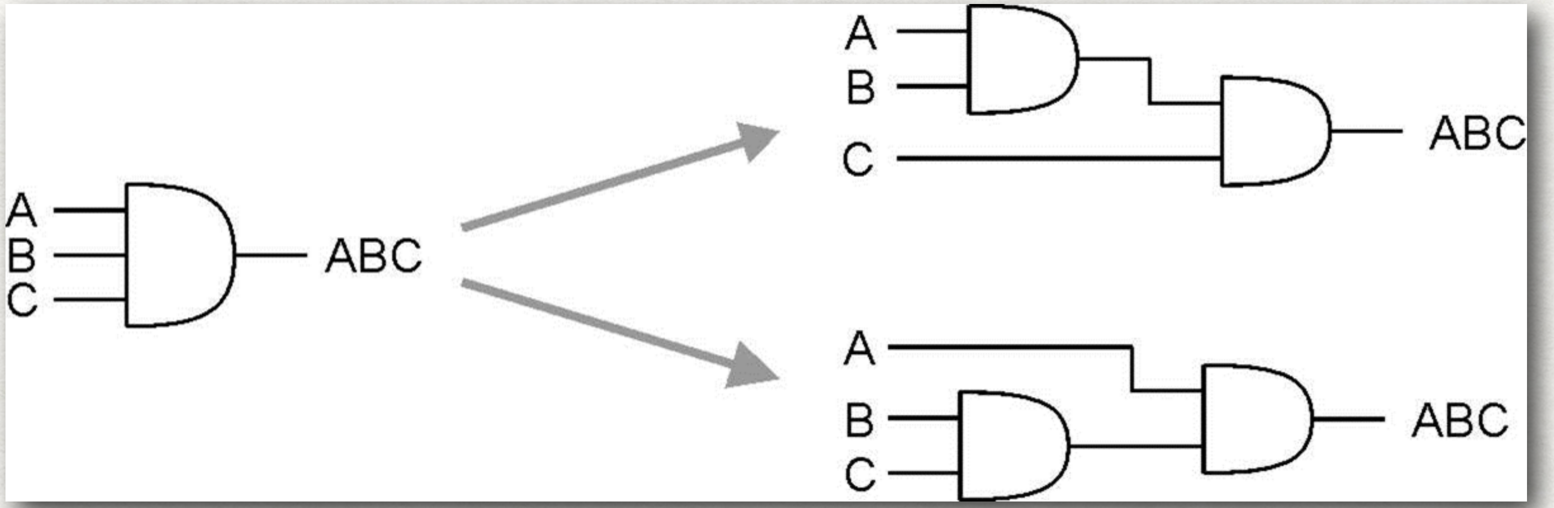
not (not A nor not B) = A nand B
another form of De Morgan's law

YOUR TURN

- Using the previous slide as a template show that we can make a NOR gate only using NAND gates
- In fact NOR gates are functionally complete by themselves
- And NAND gates are functionally complete by themselves
- But we will normally use NOT, AND and OR because it makes the design of circuits easier for our purposes

SHORTHAND MULTIPLE INPUT ANDS / ORS

- AND/OR can take any number of inputs
- AND = 1 if all inputs are 1
- OR = 1 if any input is 1
- So we can picture them like this:



BUILDING COMPUTER CIRCUITS

- To build a circuit from desired outcomes
  - Use a standard **circuit construction algorithm**
    - sum-of-products algorithm
- To convert a circuit to a Boolean expression
  - Start with output and work backward
    - Find next gate back, convert to Boolean operator
    - Repeat for each input, filling in left and right side
- To convert a Boolean expression to a circuit
  - Similar approach

BUILDING CIRCUIT EXAMPLE

Example from text (page 195)

- Build truth table (this circuit has two outputs)

Inputs			*Outputs*	
a	***b***	***c***	***Output-1***	***Output-2***
0	0	0	0	1
0	0	1	0	0
0	1	0	1	1
0	1	1	0	1
1	0	0	0	0
1	0	1	0	0
1	1	0	1	1
1	1	1	0	0

$2^3 = 8$ input combinations

SUM OF PRODUCTS

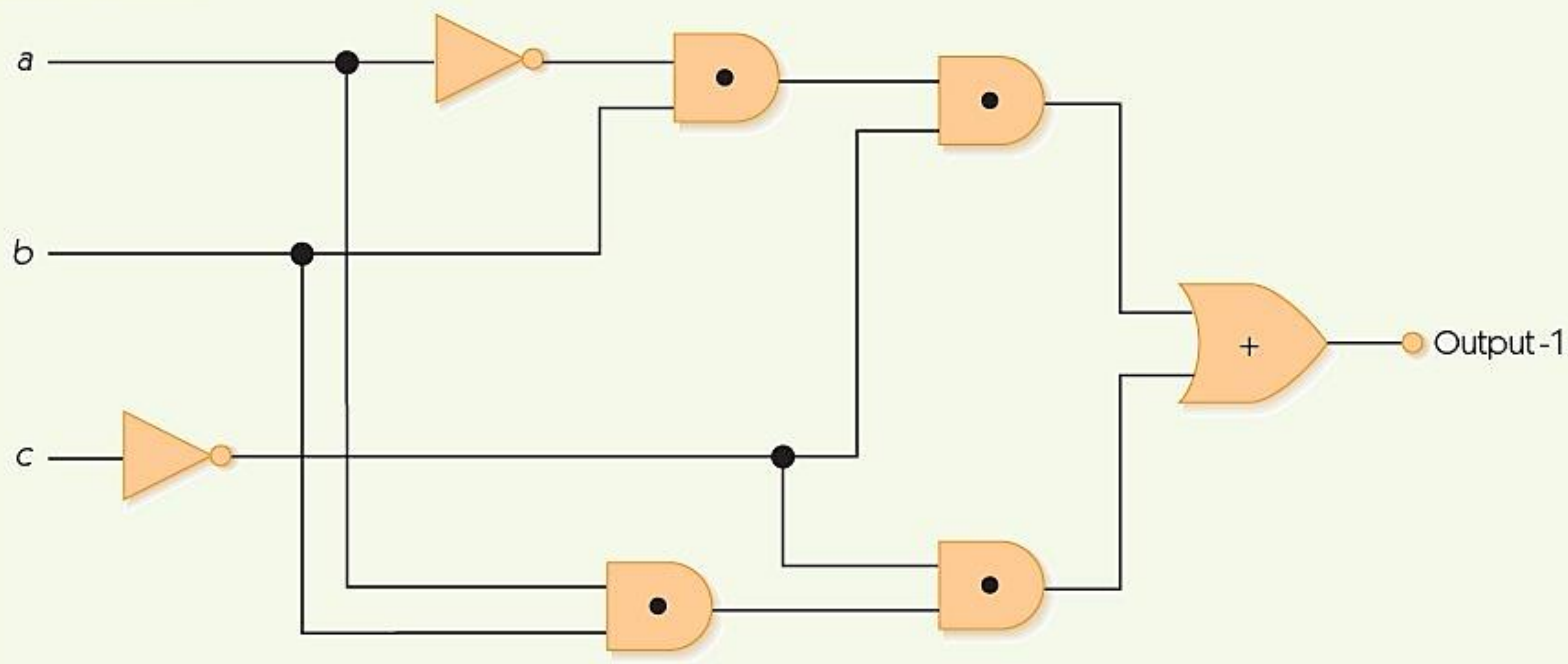
Example from text

- Find true rows for Output-1

Inputs			
a	*b*	*c*	*Output-1*
0	0	0	0
0	0	1	0
0	1	0	1 ← case 1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1 ← case 2
1	1	1	0

OR THE OUTPUT OF THE ANDS

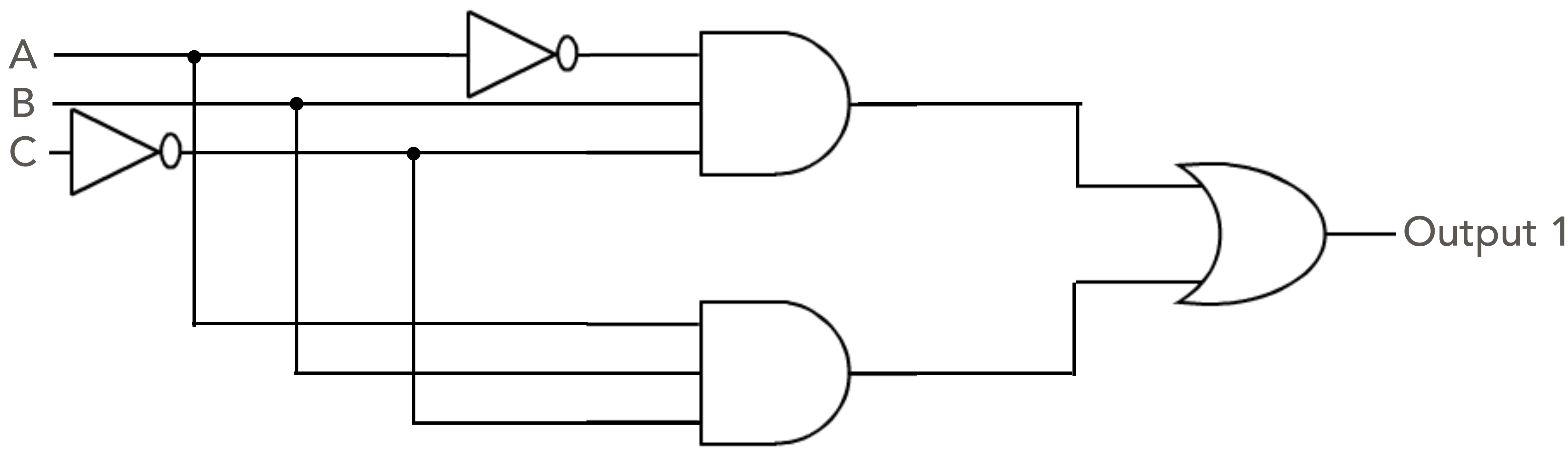
FIGURE 4.24



Circuit diagram for the output labeled Output-1

CLEARER

- It is clearer to see if we use more than one input to the ANDs
- This is the sum of products algorithm, you OR together the values coming from the ANDs. Each AND corresponds to a row with 1 in the output



THE SUM OF PRODUCTS ALGORITHM

FIGURE 4.25

1. Construct the truth table describing the behavior of the desired circuit
2. While there is still an output column in the truth table, do Steps 3 through 6
3. Select an output column
4. Subexpression construction using AND and NOT gates
5. Subexpression combination using OR gates
6. Circuit diagram production
7. Done

The sum-of-products circuit construction algorithm

YOU DO

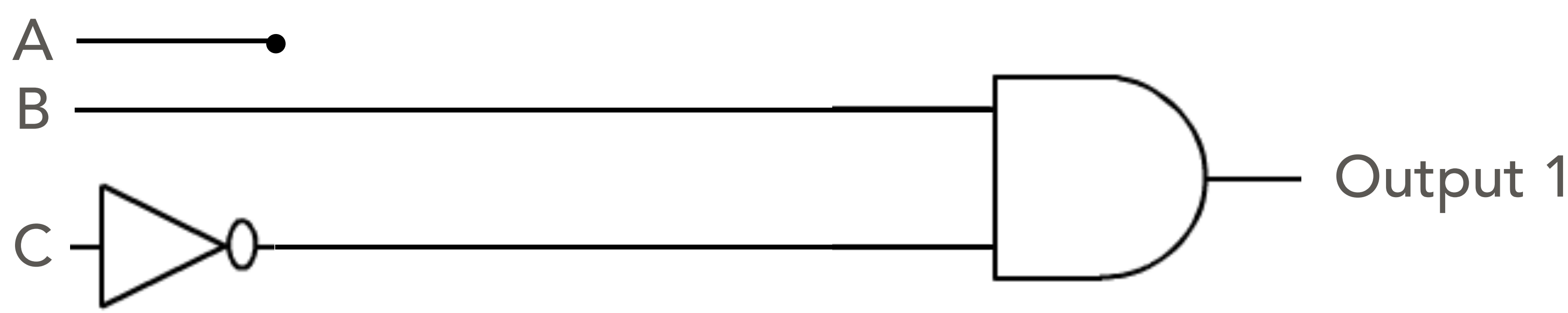
- Use the sum of products technique to produce the circuit for output-2 of this truth table.

Inputs			Outputs	
a	*b*	*c*	*Output-1*	*Output-2*
0	0	0	0	1
0	0	1	0	0
0	1	0	1	1
0	1	1	0	1
1	0	0	0	0
1	0	1	0	0
1	1	0	1	1
1	1	1	0	0

$2^3 = 8$ input combinations

CIRCUIT OPTIMISATION

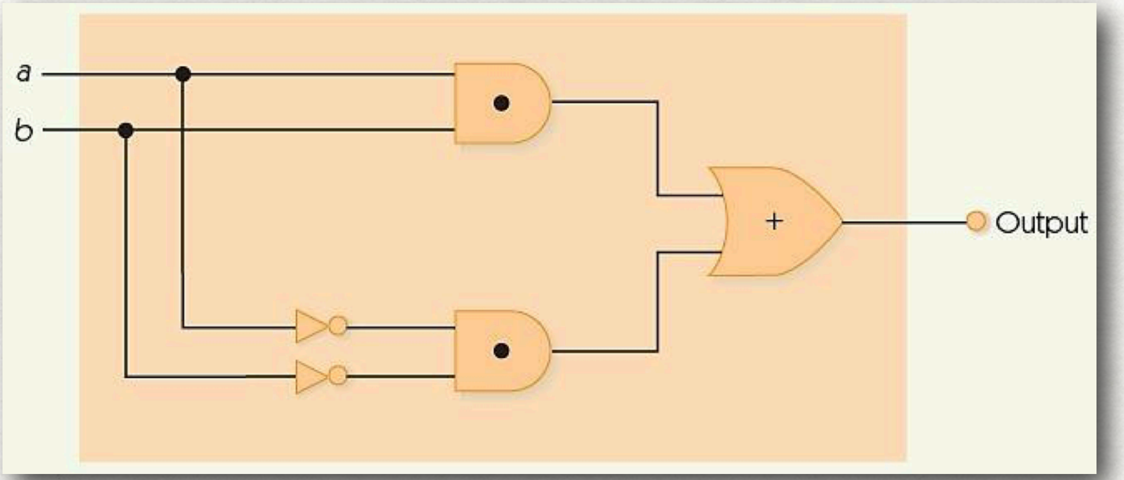
- The circuit we produced in slide 15 is not optimal
- Show that the following circuit also creates the same function
- We do not study circuit optimisation in this course - that is in PHYSICS 140



Inputs a	b	c	Output-1	
0	0	0	0	
0	0	1	0	
0	1	0	1	← case 1
0	1	1	0	
1	0	0	0	
1	0	1	0	
1	1	0	1	← case 2
1	1	1	0	

MORE FOR YOU TO DO

- Write the boolean expression equivalent to this circuit



- Draw the truth table

- Draw the circuit equivalent to the expression:
  NOT (A AND NOT B) OR A

- Using the sum of products technique draw the circuit for the truth table

A	B	Output
0	0	1
0	1	1
1	0	0
1	1	1